МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»
(БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Отчет по учебной ознакомительной практике

Выполнил студент группы ВТ-202
Скляренко Александр Викторович
_____
(подпись студента)

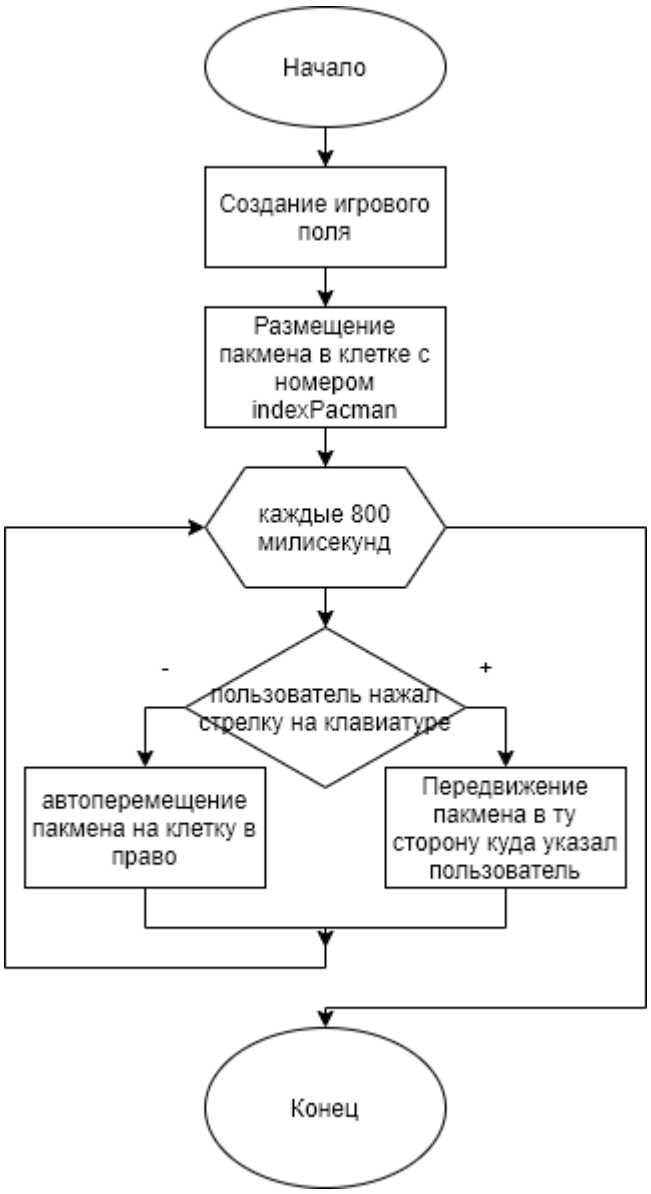
Проверил:
Бондаренко Татьяна Владимировна
_____
(подпись руководителя практики)

Оценка_____

Белгород 2021

Содержание

Блок – схема в укрупленных блоках



Задания к работе

Тема: Разработка браузерной 2D игры Pacman на языке JavaScript.

Задачи которые будут решены в ходе летней практики:

1. Ознакомиться с языком программирования JavaScript.
2. Реализовать алгоритм создания игрового поля.
3. Реализовать автоматическое движение pacman-а и считывание действий пользователя.

Не много о JavaScript

JavaScript — это кросс-платформенный, объектно-ориентированный скриптовый язык, являющийся небольшим и легковесным. Внутри среды исполнения JavaScript может быть связан с объектами данной среды и предоставлять программный контроль над ними.

JavaScript включает стандартную библиотеку объектов, например, Math, а также базовый набор языковых элементов, например, операторы и управляющие конструкции. Ядро JavaScript может быть расширено для различных целей путём добавления в него новых объектов, например:

- JavaScript на стороне клиента расширяет ядро языка, предоставляя объекты для контроля браузера и его Document Object Model (DOM). Например, клиентские расширения позволяют приложению размещать элементы в форме HTML и обрабатывать пользовательские события, такие как щелчок мыши, ввод данных в форму и навигация по страницам.
- JavaScript на стороне сервера расширяет ядро языка, предоставляя объекты для запуска JavaScript на сервере. Например, расширение на стороне сервера позволяет приложению соединяться с базой данных, обеспечивать непрерывность информации между вызовами приложения или выполнять манипуляции над файлами на сервере.

2)Не много о pacman :

Pacman аркадная видеоигра, разработанная японской компанией Namco и вышедшая в 1980 году. После выхода в Японии игра была принята хорошо, но не стала популярной. В Америке же аудитория была впечатлена отсутствием в аркаде насильственного мотива, что привлекло в том числе женскую аудиторию и помогло заработать лояльность родителей к видеоигре.

3)Правила игры :

Задача игрока — управляя Пакманом, съесть все точки в лабиринте, избегая встречи с привидениями, которые гоняются за героем. Если игрок съел все точки, то он прошел уровень, а если игрок попался приведению то он проиграл

Реализация создания игрового поля

Для начала мы должны создать HTML-страницу,где собсвенно и будет проходить все действие игры

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="style.css">
    <title>Document</title>
</head>

<body>
    <div id="game-board"></div>
    <h2>SCORE : <span id="score">0</span></h2>
    <script src="/js/ping/snake/script.js"></script>
</body>

</html>
```

В блоке div с id = game-board и будет сама игра
А в блочном элементе h2 мы будем выводить количество очков набранных пользователем

Далее нам нужно стилизовать блок для игры, для создания CSS файла я использую препроцессор Sass а именно Scss ибо как мне кажется он удобней и интуитивно понятней.

Scss код

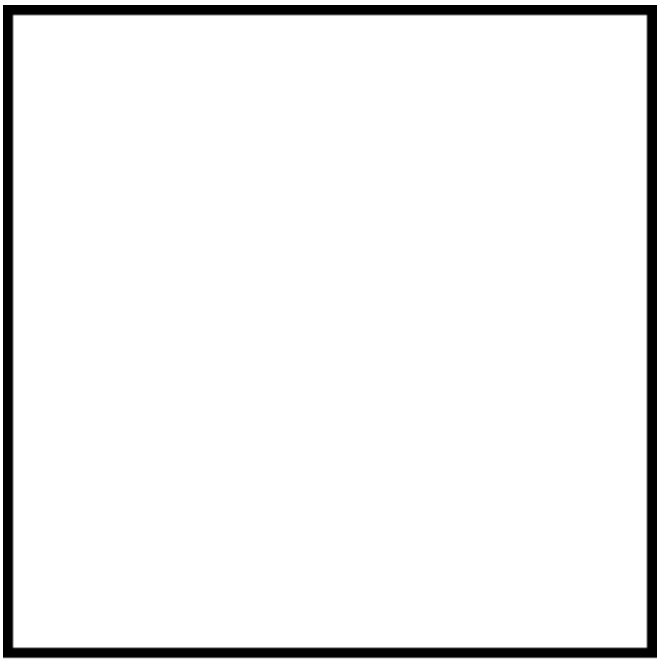
```scss
#game-board {
    width: 180px;
    height: 180px;
    display: flex;
    flex-wrap: wrap;
    border: 3px solid black;
    div {
        width: 30px;
        height: 30px;
    }
    .pacman {
        z-index: 3;
        border-radius: 50%;
        width: 30px;
        height: 30px;
    }
    .ghost {
        z-index: 5;
        width: 30px;
```

```
        height: 30px;
    }
    .point {
        z-index: 2;
        background-image: url('../snake/clipart1405756.png');
        background-size: contain;
    }
    .wall {
        background-color: black;
    }
    .empty {
        background-color: #fff;
    }
}
```

Распишем все более делательно
1) #game-board – это собственно сам блок в котором происходит игра мы задаем ему высоту и ширину по 180 пикселей, делаем границу в 3 пикселя черного цвета, и делаем flex – контейнером, и разрешаем флекс элементам разрываться если они не помещаются во flex – контейнер
Далее всем блокам div являющиеся дочерними блоку game-board мы задаем высоту и ширину в 30 пикселей это сами блоки по которым будет ходить пакмен и призраки
Далее элементам с классом pacman и ghost задаем ширину отдельно, т.к в это будут картинки а они не наследуют родительскую ширину и высоту, так же задаем свойство z-index для того что бы при переходе на клетку с поинтом пакмен и призрак перекрывали собой этот блок

Далее блоку point задаем z-index меньший чем у пакмена и призрака и стилизуем его и пустые блоки и стенки

Полученное поле :



SCORE : 0

С внешним видом закончили, теперь переходим непосредственно к программированию
Для начала нам нужно получить элементы HTML страницы
Для этого я создаю 2 константы и получаю в первую блок где будет происходить игра, а во вторую тег для вывода счетчика

```
const board = document.querySelector('#game-board');
const score = document.querySelector('#score');
```

Далее создаем 2 массива.

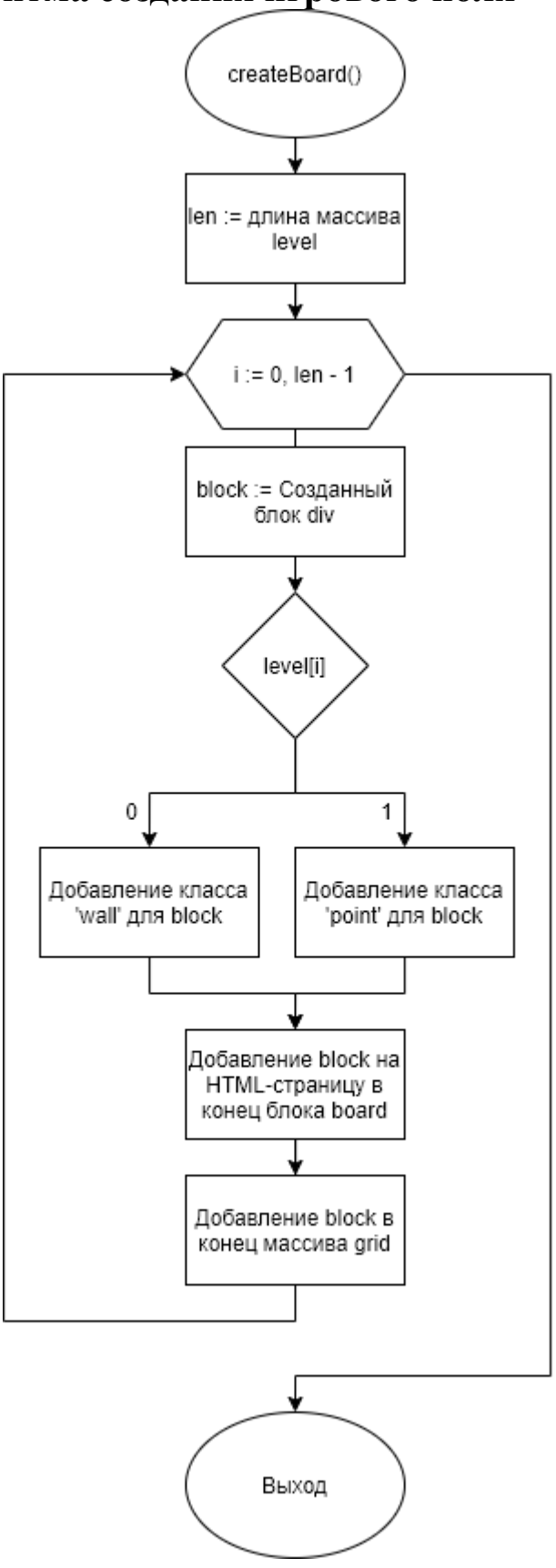
```
    const level = [
        2, 1, 1, 1, 1, 0,
        1, 1, 0, 1, 1, 0,
        1, 1, 0, 1, 1, 0,
        1, 1, 0, 1, 1, 0,
        1, 1, 0, 1, 1, 0,
    ];

    let grid = [];
```

В массиве level я храню числа от 0..1 где
0 – стенка,1- поинт
А в массив grid я буду записывать сами блоки для дальнейшей работы с ними

Функция создания игрового поля
Назначение создает игровое поле на HTML-странице

Блок-схема алгоритма создания игрового поля



```
function createBoard() {
        for (let i = 0; i < level.length; i++) {
            let block = document.createElement('div');
            switch (level[i]) {
                case 0:
                    block.classList.add('wall');
                    break;
                case 1:
                    block.classList.add('point');
                    break;
            }
            board.insertAdjacentElement(
                'beforeend',
                block,
            )
            grid.push(block);
        }
    }
```

Описание работы функции :

В цикле перебираем все элементы массива level, так же в цикле создаем блочный элемент и записываем его в переменную block и в зависимости от того какое число записано в массиве level на i-той позиции ,присваиваем block класс либо wall либо point и в конце цикла добавляем block на HTML – страницу.

Результат работы функции

Реализация движения pacman

Функция передвижения пакмена

```
let flag = false; // true если пользователь нажал клавишу
let step = 6; // шаг для пакмена если ему нужно пойти вверх или вниз
 function movePacman() {
```

```
        document.addEventListener('keydown', userMove);
        if (!flag) {
            avtoMovePacman();
        }
        flag = false;
    }
```

Разберем эту функцию более детально
В первой строке мы вызываем фукнцию userMove если пользователь нажал какую либо клавишу на клавиатуре , и если она была вызвана то flag == true и мы не зайдем в развилку тем самым не вызвав функцию avtoMovePacman

Функция userMove

```
function userMove(e) {
        flag = true;
        switch (getUserChoose(e)) {
            case 0:
                if (!grid[indexPacman + step].classList.contains('wall') && indexPac
man + step < grid.length) { // шаг вниз
                    delPacman();
                    indexPacman += step;
                    printCountPoint();
                    addPacman();
                }
                break;
            case 1:
                if (!grid[indexPacman - step].classList.contains('wall') && indexPac
man >= 6) { // шаг вверх
                    delPacman();
                    indexPacman -= step;
                    printCountPoint();
                    addPacman();
                }
                break;
            case 2:
                if (!grid[indexPacman - 1].classList.contains('wall') && indexPacman
 % step != 0) { // шаг влелво
                    delPacman();
                    indexPacman--;
                    printCountPoint();
                    addPacman();
                }
                break;
            case 3:
                if (!grid[indexPacman + 1].classList.contains('wall') && (indexPacma
n + 1) % (step) != 0) { // шаг вправо
                    delPacman();
                    indexPacman++;
                    printCountPoint();
                    addPacman();
                }
                break;
```

```
        }
    }
```

Разберем эту функцию более детально
Для начала нам надо определить какую именно клавишу нажал пользователь для этого я вызываю функцию getUserChoice(e), и в зависимости от того что она вернет я передвигаю пакмена
Если она вернула 0, то значит нужно походить вниз, сначала я проверю не является ли ход вниз стенкой, и потом проверяю не является ли ход вниз границей карты, и если оба эти условия соблюдены то я вызываю функцию printCountPoint(),которая выводит в поле SCORE количество съеденных поинтов и затем реализую **алгоритм перемещения**

Алгоритм перемещения заключается в следующем, сначала я вызываю функцию delPacman(), в которой я удаляю элемент с HTML-страницы,который имеет класс pacman, затем я увеличиваю индекс пакмена на то число, которое нужно взависимости от места куда походить (indexPacman),затем я вызываю функцию addPacman(), которая добавляет элемент с классом pacman на страницу HTML но уже по новому индексу, следовательно создается как бы иллюзия движения

Функции используемые при алгоритме перемещения:

1. getUserChoice(e)

```
function getUserChoice(e) {
        switch (e.key) {
            case 'ArrowDown':
                return 0;
            case 'ArrowUp':
                return 1;
            case 'ArrowLeft':
                return 2;
            case 'ArrowRight':
                return 3;
        }
    }
```

Назначение : В нее передается специальный объект которой возникает при нажатии клавишы клавиатуры, я обращаюсь к его полю с именем key и в зависимости чему равно это поле возвращаю число (0 – низ, 1 – верх, 2 – лево,3 – право)

2. delPacman()

```
function delPacman() {
        grid[indexPacman].querySelector('.pacman').remove();
    }
```

Назначение : Удаляет элемент с классом pacman в массиве grid по индексу indexPacman

3. addPacman

```
function addPacman() {
        grid[indexPacman].className = 'empty';
        grid[indexPacman].insertAdjacentElement(
            'afterbegin',
            pacMan,
        )
    }
```

Назначение : заменяет все классы элемента по индексу indexPacman в массиве grid на empty,затем по этому же индексу добавляет блок pacman

Функции которые не используются в алгоритме перемещения

1. **printCountPoint()**

```
function printCountPoint() {
        point += grid[indexPacman].classList.contains('point');
        score.textContent = point;
    }
```

Назначение : если элемент с индексом indexPacman в массиве grid содержит класс ”point” то point := point + 1 иначе + 0, затем вывод point на страницу

2. avtoMovePacman

```
function avtoMovePacman() {
        if (level[indexPacman + 1] != 0 && (indexPacman + 1) % (step) != 0) {
            delPacman();
            indexPacman++;
            printCountPoint();
            addPacman();
        }
    }
```

Назначение : проверяет может ли сходить пакмен влево (нет стены и не граница поля) и если может то реализуется **алгоритм перемещения**

Проверка работы перемещения пакмена

видео